Reliable Indoor Navigation on Humanoid Robots using Vision-Based Localization

Thomas Moulard, Pablo Fernández Alcantarilla, Florent Lamiraux, Olivier Stasse, Frank Dellaert

Reliable Indoor Navigation on Humanoid Robots using Vision-Based Localization

Thomas Moulard[1], Pablo F. Alcantarilla[2], Florent Lamiraux[1], Olivier Stasse[1], Frank Dellaert[3]

***Abstract*— Reliable localization on a humanoid robot is a *sine qua non* condition to succeed in realizing complex robotics scenarios. Before dealing with perturbations and online modification of the environment, one has to make sure that the planned trajectory alone will be correctly followed. This paper demonstrates that a control framework suited to humanoid robots relying on a vision-based localization system can achieve this goal. Our localization framework is based on a real-time vision-based localization system that assumes that a pre-existing 3D map of the environment exists and allows to obtain accurate results in complex robotics scenarios. By compensating for execution errors such as drifts and robot model errors, the HRP-2 robot is able to achieve high precision tasks.**

***Index Terms*— Humanoid robots, Localization, Motion Planning, Robot Control, Stereo Vision**

I. Introduction

In this paper, we consider the problem of real-time reliable trajectory execution for humanoid robots. A precise localization on a humanoid robot is both crucial to achieve reliable trajectory execution and challenging due to humanoid robots specific features. We propose a novel control architecture for humanoid robots where the current localization is used by the robot controller and planner trajectory. The objective of this work is to localize the humanoid robot HRP-2 [13] while it is navigating through an indoor environment.

Localization is crucial on a humanoid robot as its position cannot be controlled directly. Legs motion are planned under the assumption that all contacts will be perfect (i.e. no friction) during the movement. This is not the case in practice and leads to execution errors which cannot be ignored. Moreover, reactive control algorithms rely on simplified dynamics models such as the *Linear Inverted Pendulum* model [12]. These imply a gap between the generated motion and the executed one. Therefore, localization is necessary to ensure a consistent robot behavior.

However, localization on a humanoid robot is challenging. Mobile robots odometry can be computed using wheel encoders and give a reasonable hint on the current robot motion. 2D maps can also be used to simplify the navigation through indoor environments. This is not the case on a legged robot on which 3D localization is required and where no encoder based reliable odometry exists. All these reasons make localization a cornerstone of reliable trajectory execution on humanoid robots. To obtain an accurate localization of the robot in the environment, we employ fast vision-based localization techniques [2] that assume that a prior 3D map of the environment exists. By means of visibility prediction [1] a fast and robust data association between a large map of 3D points and perceived 2D features in the image can be performed very efficiently yielding an accurate pose estimate of the robot in the 3D map. This prior 3D map of the environment can be generated by standard visual Simultaneous Localization and Mapping (SLAM) techniques [6, 14].

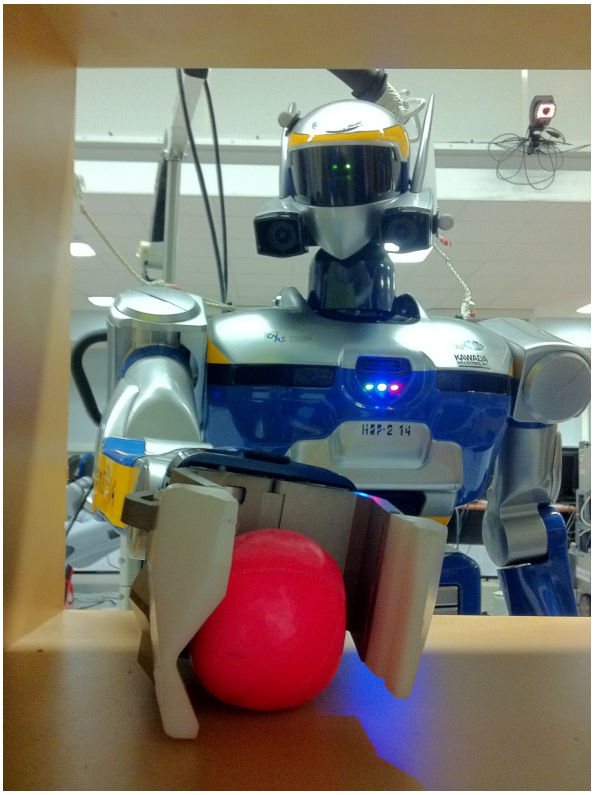

Fig. 1. HRP-2 dropping a ball on a shelf after having navigated through an indoor environment. Navigation followed by manipulation are typical scenarios where walking drift can prevent a task from being accomplished.

In this article, the past results in vision processing on humanoids robots are presented in Section II. Section III details how the vision is linked to the robot controller via our proposed control scheme. The vision-based localization algorithm is described in Section IV. The experimental results are discussed and compared to motion capture data in Section V. Finally, conclusions and future work are described in Section VI.

[1]T. Moulard, F. Lamiraux, O. Stasse are with LAAS-CNRS; 7, avenue du Colonel Roche 31077 Toulouse France thomas.moulard@gmail.com

[2] P.F. Alcantarilla is with ISIT-UMR 6284 CNRS, Université d'Auvergne, Clermont Ferrand, France pablofdezalc@gmail.com

[3] F. Dellaert is with School of Interactive Computing, Georgia Institute of Technology, Atlanta, GA 30332, USA. dellaert@cc.gatech.edu

II. Related Work

Cameras seem to be an appealing sensor for humanoid robotics applications: they are small, cheap and light-weight compared to other sensors such as laser scanners. Despite of this, there have been only limited attempts at vision-based localization for humanoid robots. Ozawa et al. [20] proposed to use stereo visual odometry to create local 3D maps for online footstep planning. The main drawback of this approach is the drift created by the accumulation of errors that typically occur in visual odometry systems [11].

Other past experiments where vision algorithms have been used to construct navigation maps for humanoid robots are [16, 17]. However, most of these works were focused on the humanoid trajectory generation replanning problem. In this paper, our approach is different: the objective is to execute a planned trajectory without any online replanning. The computation of a whole-body trajectory remains very costly and do not meet the real-time requirements of small robots such as Nao [1]. Therefore, local trajectory deformation is important to achieve high reactivity and to avoid unnecessary computations.

The work by Dune et al. [7] relies on visual servoing to compute a center of mass reference velocity to control the walking trajectory generation algorithm described in [10]. By computing the foot placement online, one can control directly the robot center of mass velocity on a 2D plane and get rid of most of the complexity of the walking process. This approach is extremely interesting, but unfortunately does not take into account obstacles in the environment. Also, this algorithm is not suited for trajectory tracking which is our objective. The following works [8, 18] aim at allowing sudden changes in the robot trajectory. Again, the objectives pursued by these works and our work differ as they alter the initial plan to react to changes in the environment.

Davison et al. [6] showed successful monocular SLAM results for small indoor environments using the HRP-2 robot. This approach, known as *MonoSLAM*, is a monocular Extended Kalman Filter (EKF) vision-based system, that allows building a small map of sparse 3D points. However, acceptable results were only obtained when the pattern generator, the robot odometry and inertial sensing were fused to aid the visual mapping into the EKF framework as it was shown in [22]. The fusion of the information from different sensors can reduce considerably the uncertainty in the camera pose and the 3D map points involved in the EKF process, yielding better localization and mapping results.

Despite of this, EKF-based approaches have important drawbacks such as the limited number of 3D points that can be tracked and divergence from the true solution due to linearization errors. As shown in [23], nonlinear optimization techniques such as bundle adjustment [19] are superior in terms of accuracy to filtering based methods, and allow tracking many hundreds of features between frames. In this paper we use the framework described in [2], where a stereo visual SLAM algorithm based on local bundle adjustment is used to compute a 3D map of the robot's environment. Then, we use the prior 3D map for an efficient data association obtaining a real-time accurate localization of the robot within the 3D environment.

[1] http://www.aldebaran-robotics.com/en/

III. Robotics Framework for Reliable Trajectory Execution

A. Notations

Before describing the robot architecture, let us introduce the following notations:

- $\mathrm{SE}(2)$ and $\mathrm{SE}(3)$ the Special Euclidian groups of dimension 2 and 3 denote rigid transformations in the 2D and 3D Euclidian spaces respectively.
- $\bar{\mathbf{q}}(t)$ denotes the robot configuration in its configuration space $\bar{\mathcal{C}}$ at time t. It is divided into a controllable part $\mathbf{q}(t) \in \mathcal{C}$ and the robot position $\mathbf{x}(t) \in \mathrm{SE}(3)$. The robot position is defined as the position of a particular robot body. In this paper, we choose the left ankle as the reference body used to compute the robot position. $\dot{\mathbf{q}}(\mathbf{t})$ describes the joint velocities.
- $\mathbf{x}_{\text{ref}}(t) \in \mathrm{SE}(3)$ is the robot planned trajectory.
- $\hat{\mathbf{x}}(t) \in \mathrm{SE}(3)$ is the robot trajectory as perceived by the localization system.
- $\mathbf{c}(t) \in \mathbb{R}^2$ and $\mathbf{z}(t) \in \mathbb{R}^2$ are respectively the projection of the robot center of mass on the floor ($z = 0$) and the Zero Momentum Point position at time t.
- $\gamma_{\text{la}}(t) \in \mathrm{SE}(3)$, $\gamma_{\text{ra}}(t) \in \mathrm{SE}(3)$ denote respectively left ankle and right ankle trajectories.
- $\theta(t) \in \mathrm{SE}(6)$ is the estimated camera pose (translation and rotation) at time t.

The set of trajectories γ_{la}, γ_{ra} and $\mathbf{c}$ defines a complete balanced walking movement and is denoted $\mathbf{\Gamma}$. It can be completed by an additional trajectory which defines the upper-body configuration during the walk. Changes in the upper-body posture do not impact the walking movement as long as the center of mass trajectory is unchanged and the induced angular momenta variations remain small. Therefore, this article will not mention upper-body trajectory although the final experiment contains arms and head movement.

B. Robotics Architecture

The proposed architecture can be divided in two parts:

1) The vision processing and localization components which are responsible for acquiring data from the robot's stereo rig system, computing an estimation of the camera pose and deducing the robot localization.
2) The control system which follows the planned trajectory while closing the loop on the localization data.

These two systems are running on two different computers embedded on the robot. They are composed of several robotics components running simultaneously. Fig. 2 illustrates the complete robotics infrastructure. To communicate, we rely heavily on both OpenHRP and ROS middlewares.

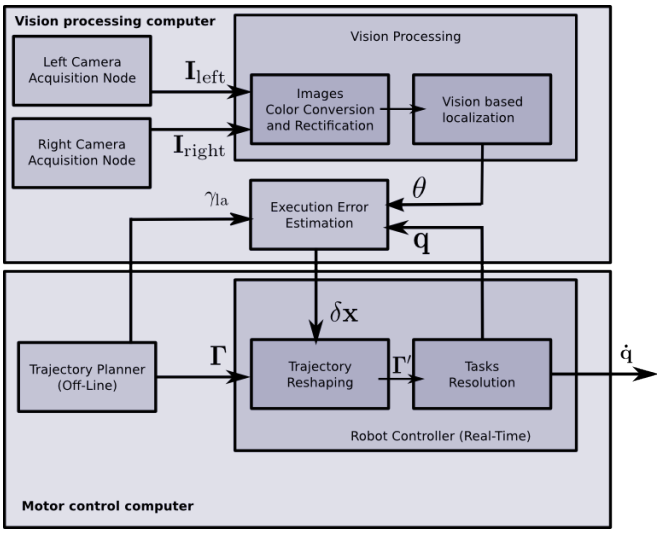

Fig. 2. Complete robotics framework overview. Let $\mathbf{I}_{\text{left}}$, $\mathbf{I}_{\text{right}}$ be respectively the left and right camera images. T_{cam}, $\hat{T}_{\text{cam}}$ respectively the planned and estimated camera position. γ the concatenation of the left ankle, right ankle, upper body and center of mass trajectories. γ' the trajectories reshaped by taking into account execution errors.

a) Vision-Based Localization: The vision system relies on two Flea2 FireWire cameras. They have been set up to grab monochrome images synchronously at 30 Hz. The image resolution is 320×240. We use the two cameras that are attached to the ears of the HRP-2 robot. These two cameras have a baseline of approximately 14.4 cm and an horizontal field of view of 90° for each of the cameras

Images are transmitted to the vision component. It is in charge of image rectification and current camera pose computation. This process is described in detail in Section IV. During our experiments, the pose estimation rate was around 16 Hz.

b) Error Estimation: The error estimation component computes $\delta\mathbf{x} \in \text{SE}(3)$, the transformation from the perceived robot position to the planned robot position and is defined at time t as:

$$\delta\mathbf{x}_t = \mathbf{x}_t.\hat{\mathbf{x}}_t^{-1} \tag{1}$$

However, $\hat{\mathbf{x}}_t^{-1}$ is not directly provided by the localization which estimates the camera pose and not the robot pose. The transformation from the camera to the left ankle (defining the robot position) can be easily deduced from the control and is not subject to significant execution errors as joint encoders allow low-level reliable servoing.

Additionally, the control is used to help the camera pose estimation. Indeed, to estimate a 3D pose, it is necessary to estimate six parameters; three for the translation: x, y, z and three for the rotation: roll, pitch, and yaw. Nevertheless, we know for sure that the camera height z and two of the three parameters of the rotation cannot drift nor have significant execution errors while walking on a flat ground. This explains why, in practice, only three parameters are estimated using the vision components whereas the z, roll and pitch parameters are known from the plan.

c) Closed-Loop Trajectory Following: To achieve walking and compute motor control, the task based control framework described in [15] is used. One interesting feature of this approach is that we can directly add tasks to follow ankles and center of mass reference trajectories. The inverse kinematics computation will then be implicitly realized by the task resolution.

Regarding trajectory generation, we use the planner described in [5] to generate the required reference trajectories.

Our control scheme adds a step which processes the reference trajectories to compensate for execution errors based on the error estimation.

Let us consider that at time t, the robot is on double support (i.e. both feet on the ground) and that an estimation of the execution error $\delta\mathbf{x}(t)$ is available. We have to alter the trajectories of the left ankle, right ankle and center of mass simultaneously while preserving robot balance. As we cannot change a foot position while it is on contact with the floor, we have to modify the trajectories during the two following steps to be able to correct both feet trajectories. To do so, we will sum each trajectory with a trajectory dependent third order polynomial denoted by $\mathbf{\Delta}_{\text{la}} \in \text{SE}(3)$, $\mathbf{\Delta}_{\text{ra}} \in \text{SE}(3)$, $\mathbf{\Delta}_{\mathbf{c}} \in \mathbb{R}^2$:

$$\begin{aligned} \gamma'_{\text{la}}(t) &= \mathbf{\Delta}_{la}(t).\gamma'_{\text{la}}(t) \\ \gamma'_{\text{ra}}(t) &= \mathbf{\Delta}_{ra}(t).\gamma'_{\text{ra}}(t) \\ \mathbf{c}'(t) &= \mathbf{\Delta}_{c}(t) + \mathbf{c}(t) \end{aligned} \tag{2}$$

Let t_1 be the end of the next step and t_2 the end of the second next step. If the first step is realized with the left foot (i.e. the right foot is the support foot). In Eq. 2, if Δ provides a correction from t_{start} to t_{end}, we consider the following constraints:

$$\begin{aligned} \mathbf{\Delta}(t) &= \begin{cases} 0., & \text{if } t \leq t_{\text{start}} \\ \delta\mathbf{x}, & \text{if } t \geq t_{\text{end}} \end{cases} \\ \dot{\mathbf{\Delta}}(t_1) &= \dot{\mathbf{\Delta}}(t_2) = 0 \end{aligned} \tag{3}$$

The constraints expressed in Eq. 3 ensures us that the correction is smooth: zero velocity at the beginning and end of the correction and that the error is correctly compensated. They are independent of the corrected trajectory (ankles or center of mass). These four constraints fully determine the four coefficients of the polynomial.

The starting and timing correction time depends on the corrected trajectory:

center of mass ($\mathbf{\Delta}_{\mathbf{c}}(t)$):	$t_{\text{start}} = t,\ t_{\text{end}} = t_1$
left ankle ($\mathbf{\Delta}_{\text{la}}(t)$):	$t_{\text{start}} = t,\ t_{\text{end}} = t_1$
right ankle ($\mathbf{\Delta}_{\text{ra}}(t)$):	$t_{\text{start}} = t_1,\ t_{\text{end}} = t_2$

The remaining issue is ensuring that the updated trajectory will be balanced. A classical approach is using the *Zero Momentum Point* (ZMP): this virtual point acts as a criterion to determine whether a trajectory is balanced or not. If it lies in the convex hull of the robot contact points all the time, the trajectory is balanced, otherwise it is not. The ZMP is defined as:

$$\mathbf{z} = \mathbf{c} + \frac{1}{m(\ddot{c}_z + g)} \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \end{pmatrix} \dot{\mathbf{L}} - \frac{c_z}{\ddot{c}_z + g} \ddot{\mathbf{c}} \tag{4}$$

where c_z is the height of the center of mass with respect to the ground, m is the mass of the robot, g is the gravity constant, $\mathbf{c} = (c_x, c_y)$ is the projection of the center mass of the robot on the ground and $\mathbf{L}$ is the angular momentum of the robot about the center of mass.

Solving this equation is time consuming as it is nonlinear and requires dynamics computation. Under the asumption that L and c_z is constant in time [12], we can obtain the follwing simplified linear model:

$$\mathbf{z} = \mathbf{c} - \frac{\mathbf{c}_z}{g}.\ddot{\mathbf{c}} \tag{5}$$

Considering $\mathbf{r}$ a polynomial depending only of $\mathbf{z}$, (V_x, V_y, W_x, W_y) free parameters used to constrain the initial position and velocity of the center of mass, a general solution of Eq. 5 is:

$$\mathbf{c}(t) = \cosh(\sqrt{\frac{g}{z_c}}.t).\mathbf{V} + \sinh(\sqrt{\frac{g}{z_c}}.t).\mathbf{W} + \mathbf{r}(t) \tag{6}$$

Given the formulation in Eq. (6), it is possible to continuously modify the center of mass trajectory to make it follow $\mathbf{z}'(t)$ the corrected trajectory. This new trajectory can be expressed as the sum of two polynomials:

$$\mathbf{c}'(t) = \cosh(\sqrt{\frac{g}{z_c}}.t).\mathbf{V} + \sinh(\sqrt{\frac{g}{z_c}}.t).\mathbf{W} + \mathbf{r}(t) + \mathbf{\Delta}(t) \tag{7}$$

It appears then that moving the center of mass final position of $\delta\mathbf{x}$ is equivalent to moving the ZMP final position of $\delta\mathbf{x}$ then solving the linear differential equation. As the flying foot end position is transformed similarly, the ZMP will stay in the convex hull contact points for any correction value $\delta\mathbf{x}$. The only limitations are feet velocities limits and assumptions realized when using the simplified linear model.

To correct properly the robot trajectory, a correction of about two to three cm every two steps is sufficient. In this case, the simplified linear model is particularly well suited as the dynamical effects of such changes are so low they can be safely ignored.

When a correction is completely applied, i.e. $t > t_2$, a new correction is computed using the current estimation of the execution error.

By combining vision-based localization, error estimation and closed-loop trajectory following, our control framework can reshape walking trajectories on the fly to reliably track the planned trajectory. We will focus the discussion in the next section on how to localize the camera precisely.

IV. Vision Processing

Our computer vision module comprises two different stages: first, a stereo visual SLAM module for building a persistent 3D map of the environment and second a vision-based localization framework with visibility prediction that assumes that a prior 3D map of the environment is given. Prior to any SLAM or localization processing, we correct the distortion of the images and perform stereo rectification. Stereo rectification simplifies considerably the stereo correspondences problem and disparity maps can be easily obtained with the rectified images. More details of the visual SLAM and vision-based localization algorithms can be found in [2]. In this section, we will briefly describe the main components of our localization framework assuming that we already have a prior 3D map of the environment. Given a prior 3D map of the environment we can use the computed map for fast and robust localization. In this context we employ the visibility prediction technique described in [1, 2] to perform an efficient data association between known 3D points and detected 2D features.

Our localization framework is composed of two different modules: initialization and a combination of vision-based localization with visibility prediction and stereo visual odometry.

A. Initialization and Re-Localization

At this stage, the robot is lost and can be located in any area of the map. Therefore, we need to find an initial camera pose to start the vision-based localization algorithm. For this purpose we detect 2D features in the new image using the Harris corner detector [9] at different scale levels and compute an appearance descriptor for each detected corner. For this purpose, we compute the appearance descriptors of the detected 2D features in the new image and match this set of descriptors against the set of descriptors from the list of stored keyframes from the prior 3D reconstruction. Similar to Speeded Up Robust Features (SURF) [3], for a detected feature at a certain scale, we compute a unitary descriptor vector of dimension 16 in order to speed up the descriptor computation. We use the upright version of the descriptors (no invariance to rotation) since upright descriptors perform better in scenarios where the camera only rotates around its vertical axis, which is often the case for humanoid robots, and are also faster to compute than its rotation invariant counterpart.

In the matching process between the camera frame and the list of keyframes, we perform a RANSAC [4] procedure forcing epipolar geometry constraints. We recover the camera pose from the stored keyframe that obtains the highest ratio of inliers. If this ratio is lower than a certain threshold, we do not initialize the localization algorithm until the robot moves into a known area yielding a higher ratio.

B. Localization

Given a prior map of 3D points and perceived 2D features in the image, the problem to solve is the estimation of the camera pose with respect to the world coordinate frame. Once the system has a good initialization, the vision-based localization system works through the following steps:

1) While the robot is moving, the stereo pair acquires a new set of images which are rectified. Then, from the rectified images, a disparity map is computed.
2) A set of image features $Z_t = \{z_{t,1} \dots z_{t,n}\}$ is detected by Harris corner detector only for the left image. Then, a feature descriptor is computed for each of the detected features.

3) Then, by using the visibility prediction algorithm, a promising subset of highly visible 3D map points is chosen and re-projected onto the image plane based on the estimated previous camera pose θ_{t-1} and known camera parameters.
4) Afterwards, a set of putative matches C_t is formed where the i-th putative match $C_{t,i}$ is a pair $\{z_{t,k}, x_j\}$ which comprises a detected feature z_k and a map element x_j. A putative match is created when the Euclidean distance between the appearance descriptors of a detected feature and a re-projected map element is lower than a certain threshold.
5) Finally, we solve the pose estimation problem minimizing the following cost error function, given the set of putative matches C_t:

$$\arg\min_{R,\mathbf{t}} \sum_{i=1}^{m} \|z_i - K\left(R \cdot x_i + \mathbf{t}\right)\|_2 \qquad (8)$$

where $z_i = (u_L, v_L)$ is the 2D image location of a feature in the left camera, x_i represents the coordinates of a 3D point in the global coordinate frame, K is the left camera calibration matrix, and R and t are respectively the rotation and the translation of the left camera with respect to the global coordinate frame. The pose estimation problem is formulated as a nonlinear least squares procedure using the Levenberg-Marquardt algorithm. The set of putative matches may contain outliers, therefore RANSAC is used in order to obtain a robust model free of outliers.

There can be some frames where the pose estimation problem cannot be solved efficiently since we may have textureless areas or slightly different viewpoints from the ones captured at the mapping sequence. For those situations, we employ stereo visual odometry to update the pose of the robot with respect to the map coordinate frame. Notice here that our system does not suffer from the typical drift of visual odometry systems, since in the next frame the system will try to localize with respect to the prior 3D map. When the number of consecutive frames where the pose estimation fails is higher than a fixed threshold (e.g. 100 frames), we declare that the tracking is lost and start a re-localization process.

V. Experimental Results

A. Experimental Setup

The experiment demonstrates that by localizing the robot while walking, HRP-2 can reach a goal position independently of the execution errors. In the chosen scenario, the robot must drop a ball on a shelf after walking 2 m. A more precise description of the scenario is depicted by Fig. 4. Empirically, we estimated the HRP-2 mean drift to be around one cm per step. Considering that 32 steps are required to reach the goal without hitting the obstacles, the usual drift would prevent the task from being accomplished. The camera trajectory estimated by the localization algorithm is validated using motion capture data. Markers have been placed on both the robot head and left ankle to provide ground truth.

Finally, the camera position is given by the algorithm described in detail in the previous sections. The localization

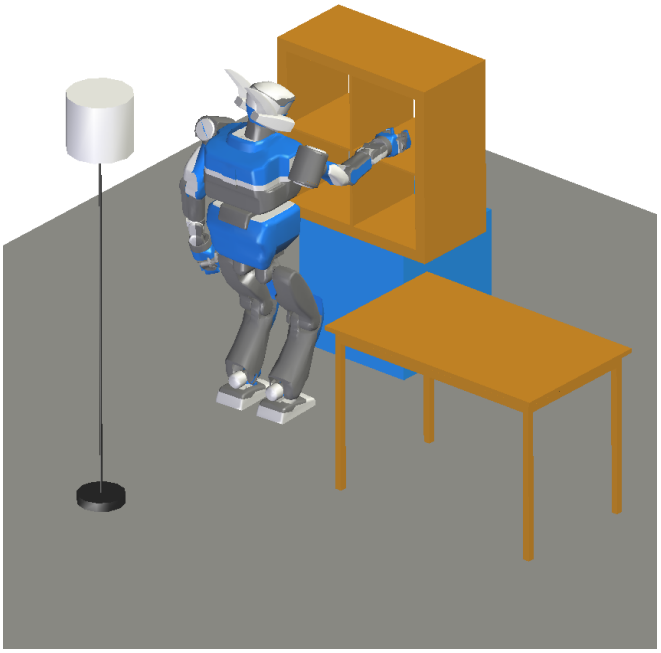

Fig. 3. The HRP-2 robot drops a ball on a shelf while localizing itself using vision based localization.

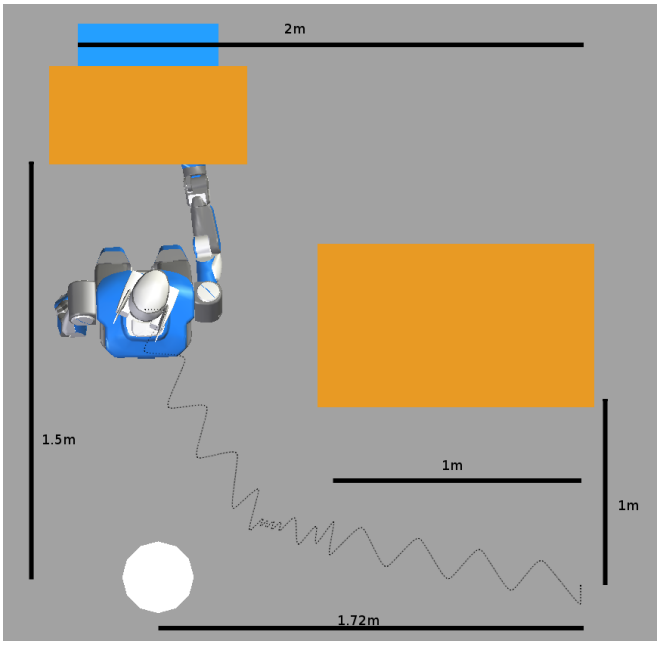

Fig. 4. Experimental setup description (top view). The dotted line displays the robot waist trajectory.

module has been running at a mean rate of 16Hz during the experiments. The vision computer running the vision based localization node is a Intel®Core™ 2 CPU T7200 @ 2.00GHz with 2Gb. of RAM.

During the experiments, the robot sometimes failed to drop the ball properly. It was either due to a bad initialization which sometimes lead to localization failure or to robot flexibility. Indeed, the shelves is quite narrow and the robot ankles

passive joints were generating hands movements which were causing collisions, even though the robot localization is precise enough. It would be interesting in a future work to not only localize the robot with respect to an absolute frame but also switch to a task-based localization at the end to correct the hand trajectory and avoid collisions.

On our platform, the vision localization module returned a pose at around 16 Hz. On the opposite, the HRP-2 robot uses a control loop at 200 Hz. To preserve real-time, the real-time component was communicating the robot configuration asynchronously at around 75 Hz. The error estimation components was running at a mean rate of 10 Hz. As a correction is added every two steps, it was not useful to evaluate the error at a higher frequency.

B. Comparison to Motion Capture Data

Fig. 5 contains the camera position estimated by the motion capture system and by the localization algorithm. The mean error is 0.2 m in translation. Drop in the algorithm precision can occur when the robot enters a part of the map which is not dense enough or where the environment does not incorporate enough texture to detect enough interest points. It happens in our scenario near the end when the camera is too close to the shelf to perceive its environment correctly.

The used motion capture system is a Motion Analysis system relying on six Eagles and four Hawks cameras. It provides an estimation of the camera position at 200 Hz with a precision error of less than one cm.

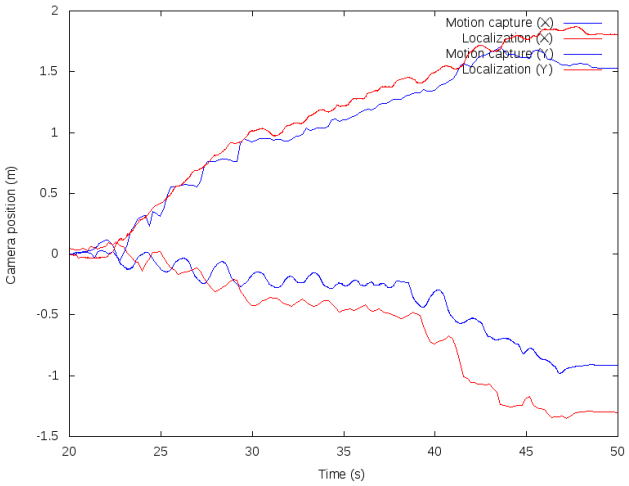

Fig. 5. Camera position estimated by both the motion capture system and the localization algorithm.

VI. Future Work and Conclusions

In this article, we demonstrated that vision-based localization allows humanoid robots to achieve complex tasks. By integrating the vision algorithm into the control framework, we have been able to gradually reshape the trajectory in order to compensate for execution errors. This allowed the robot to achieve complex tasks which would be difficult to realize otherwise.

However, several intesresting additions remain to be integrated. First, the local trajectory modifications are not checked to make sure that no auto-collision occurs. Currently, a conservative maximum correction (i.e. 2 cm every two steps) is imposed for safety. By applying recent works such as fast feasibility tests described by Perrin et al. [21], we would be able to increase the maximum correction without compromising the robot safety. Second, the HRP-2 robot embeds an Inertial Measurement Unit which could be used to estimate the robot chest attitude. It would definitely be interesting to provide an initial estimation of the robot motion to help the localization process. In addition, we are interested in improving the capabilities of our visual SLAM and vision-based localization systems towards the goal of long-term localization to deal with possible changes in the environment.

Acknowledgments

References

[1] P.F. Alcantarilla, K. Ni, L.M. Bergasa, and F. Dellaert. Visibility Learning in Large-Scale Urban Environment. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, Shanghai, China, 2011.

[2] P.F. Alcantarilla, O. Stasse, S. Druon, L.M. Bergasa, and F. Dellaert. How to localize humanoids with a single camera? *Autonomous Robots*, 2012.

[3] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. SURF: Speeded Up Robust Features. *Computer Vision and Image Understanding*, 110(3):346–359, 2008.

[4] R. Bolles and M. Fischler. A RANSAC-based approach to model fitting and its application to finding cylinders in range data. In *Intl. Joint Conf. on AI (IJCAI)*, pages 637–643, Vancouver, Canada, 1981.

[5] S. Dalibard, A. El Khoury, F. Lamiraux, M. Taix, and J.-P. Laumond. Small-space controllability of a walking humanoid robot. In *IEEE-RAS Intl. Conference on Humanoid Robots*, pages 739–744, Bled, Slovenia, 2011.

[6] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse. MonoSLAM: Real-Time Single Camera SLAM. *IEEE Trans. Pattern Anal. Machine Intell.*, 29(6):1052–1067, 2007.

[7] C. Dune, A. Herdt, O. Stasse, P.-B. Wieber, K. Yokoi, and E. Yoshida. Cancelling the sway motion of dynamic walking in visual servoing. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 3175 –3180, oct. 2010. doi: 10.1109/IROS.2010.5649126.

[8] K. Harada, S. Kajuta, K. Kaneko, and H. Hirukawa. An analytical method on real-time gait planning for a humanoid robot. In *IEEE-RAS Intl. Conference on Humanoid Robots*, pages 640–655, Los Angeles, CA, USA, 2004.

[9] C. Harris and M. Stephens. A combined corner and edge detector. In *Proc. Fourth Alvey Vision Conference*, pages 147–151, 1988.

[10] A. Herdt, H. Diedam, P.-B. Wieber, D. Dimitrov, K. Mombaur, and M. Diehl. Online Walking Motion

Generation with Automatic Foot Step Placement. *Advanced Robotics*, 24(5-6):719–737, 2010.

[11] M. Kaess, K. Ni, and F Dellaert. Flow separation for fast and robust stereo odometry. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, Kobe, Japan, 2009.

[12] S. Kajita, F. Kanehiro, K. Kaneko, K. Yokoi, and H. Hirukawa. The 3D linear inverted pendulum mode: a simple modeling for a biped walking pattern generation. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 239–246, Maui, HI, USA, 2001.

[13] K. Kaneko, F. Kanehiro, S. Kajita., H. Hirukawa, T. Kawasaki, M. Hirata, K. Akachi, and T. Isozumi. Humanoid robot HRP-2. In *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*, volume 2, pages 1083 – 1090 Vol.2, 26-may 1, 2004. doi: 10.1109/ROBOT.2004.1307969.

[14] K. Konolige and M. Agrawal. FrameSLAM: from bundle adjustment to real-time visual mapping. *IEEE Trans. Robotics*, 24(5):1066–1077, Oct 2008.

[15] N. Mansard, O. Stasse, P. Evrard, and A. Kheddar. A versatile Generalized Inverted Kinematics implementation for collaborative working humanoid robots: The Stack Of Tasks. In *Intl. Conf. on Advanced Robotics (ICAR)*, pages 1–6, Munich, Germany, 2009.

[16] P. Michel, J. Chestnutt, J. Kuffner, and T. Kanade. Vision-guided humanoid f ootstep planning for dynamic environments. In *IEEE-RAS Intl. Conference on Humanoid Robots*, pages 13–18, Tsukuba, Japan, 2005.

[17] P. Michel, J. Chestnut, S. Kagami, K. Nishiwaki, J. Kuffner, and T. Kanade. Online environment reconstruction for biped navigation. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 3089–3094, Orlando, FL, USA, 2006.

[18] M. Morisawa, K. Harada, S. Kajita, S. Nakaoka, K. Fujiwara, F. Kanehiro, K. Kaneko, and H. Hirukawa. Experimentation of humanoid walking allowing immediate modification of foot place based on analytical solution. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 3989 –3994, april 2007. doi: 10.1109/ROBOT.2007.364091.

[19] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, and P. Sayd. Generic and Real-Time Structure from Motion using Local Bundle Adjustment. *Image and Vision Computing*, 27(8):1178–1193, 2009.

[20] R. Ozawa, Y. Takaoka, Y. Kida, K. Nishiwaki, J. Chestnutt, and J. Kuffner. Using visual odometry to create 3D maps for online footstep planning. In *IEEE Intl. Conference on Systems, Man and Cybernetics*, pages 2643–2648, Hawaii, USA, 2005.

[21] N. Perrin, O. Stasse, F. Lamiraux, and E. Yoshida. Approximation of Feasibility Tests for Reactive Walk on HRP-2. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 4243–4248, Anchorage, AK, USA, 2010.

[22] O. Stasse, A.J. Davison, R. Sellaouti, and K. Yokoi. Real-time 3D SLAM for a humanoid robot considering pattern generator information. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 348–355, Beijing, China, 2006.

[23] H. Strasdat, J.M.M. Montiel, and A.J. Davison. Real-time Monocular SLAM: Why Filter? In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 2657–2664, Anchorage, AK, USA, 2010.